We want to solve the following problem:

Input: A sequence of n numbers $a_1, a_2, \ldots, a_n$

Output: A reordering (permutation) $a_1', a_2', \ldots, a_n'$ of the input sequence such that $a_1' \leq a_2' \leq \ldots \leq a_n'$

Example: Input: 8, 2, 4, 9, 3, 6

Output: 2, 3, 4, 6, 8, 9.

Insertion Sort

Input: 8 2 4 9 3 6 → 8 8 4 9 3 6

After 1st iteration: 2 8 4 9 3 6 → 2 8 8 9 3 6

2nd iteration: 2 4 8 9 3 6

3rd iteration: 2 4 8 9 3 6 → 2 4 8 9 9 6

2 4 8 8 9 6

2 4 4 8 9 6

4th iteration 2 3 4 8 9 6

5th iteration 2 3 4 8 6 9

InsertionSort

Input: Array A of n numbers

Output: A sorted.

```
for k = 1 to n-1
    key = A[k]
    i = k-1
    while i ≥ 0 && A[i] > key
        A[i+1] = A[i]
        i = i-1
    A[i+1] = key
return A.
```

Correctness:

Loop invariant: (Property of a loop that is true before or after each iteration)

→ After the k-th execution of the loop $A[0 \ldots k]$ is sorted.

Proof: . $A[0]$ is sorted (base case)

. If $A[0 \ldots k-1]$ is sorted, after inserting $A[k]$, we have that $A[0 \ldots k]$ is sorted. ∎

Consequently: $A[0 \ldots n-1]$ is sorted when the algorithm finishes.

Running Time:

```
for k = 1 to n-1
    key = A[k]
    i = k-1
    while i >= 0 && A[i] > key
        A[i+1] = A[i]
        i = i-1.
    A[i+1] = key
return A.
```

Cost	Times
c_1	n
c_2	$n-1$
c_3	$n-1$
c_4	$\sum_{k=1}^{n-1} t_k$
c_5	$\sum_{k=1}^{n-1}(t_k-1)$
c_6	$\sum_{k=1}^{n-1}(t_k-1)$
c_7	$n-1$.
c_8	1

$t_k :=$ # of times comparisons in while loop is evaluated.

$$\underbrace{\text{Running Time}} = c_1 \cdot n + (c_2 + c_3 + c_7)(n-1) + c_4 \cdot \sum_{k=1}^{n-1} t_k + (c_5 + c_6) \cdot \sum_{k=1}^{n-1} (t_k - 1) + c_8$$

↳ Depends on the size of the input. So, let **Running Time = T(n).**

T(n) may depend also on the input itself!

If A is sorted, then key ≥ A[i] so $t_k = 1$ for $k = 1, \dots, n-1$.

$$T(n) = c_1 \cdot n + (c_2 + c_3 + c_7)(n-1) + c_4(n-1) + c_8$$

$$T(n) = \underbrace{(c_1 + c_2 + c_3 + c_4 + c_7)}_{a} \cdot n - \underbrace{(c_2 + c_3 + c_7 + c_4 - c_8)}_{b}$$

↳ **Linear !!!...**

If A is in reverse order, then A[k] will be compared (in the while loop) to A[0], A[1], ..., A[k-1] so that $t_k = k$. for $k = 1, \dots, n-1$

$$T(n) = c_1 \cdot n + (c_2 + c_3 + c_7)(n-1) + c_4 \cdot \sum_{k=1}^{n-1} k + (c_5 + c_6) \cdot \sum_{k=1}^{n-1} (k-1) + c_8$$

$$\sum_{k=1}^{n-1} k = \frac{n(n-1)}{2} \qquad \sum_{k=1}^{n-1} (k-1) = \frac{n(n-1)}{2} - (n-1) = (n-1)\left[\frac{n}{2} - 1\right] = \frac{(n-1)(n-2)}{2}$$

$$T(n) = (c_1 + c_2 + c_3 + c_7)n - (c_2 + c_3 + c_7) + c_4 \cdot \frac{n(n-1)}{2} + (c_5 + c_6)\frac{(n-1)(n-2)}{2} + c_8$$

$$T(n) = \left(\frac{c_4}{2} + \frac{c_5}{2} + \frac{c_6}{2}\right)n^2 + \left(c_1 + c_2 + c_3 + c_7 - \frac{c_4}{2} - 3\frac{c_5}{2} - \frac{3c_6}{2}\right)n + c_5 + c_6 - c_2 - c_3 - c_7 + c_8$$

T(n) is Quadratic!!

Is this the worst case? Yes!! Since $T_k \leq k$ because i is allways decreased inside the while loop.